

MUR-100 读卡器使用指南

USB 接口的卡片读写器

UM06540101 V2.60 Date: 2009/08/11 产品用户手册

类别	内容
关键词	MUR-100 卡片读写器 ISO14443-4
摘 要	支持 ISO14443-4 协议的 MUR-100 卡片读写器使用指南

广州致远电子有限公司



修订历史

版本	日期	原因
V2.60	2009/08/11	创建文档

销售与服务网络（一）

广州周立功单片机发展有限公司

周立功单片机 http://www.zlgmcu.com

地址：广州市天河北路 689 号光大银行大厦 12 楼 F4
邮编：510630
电话：(020)38730916　38730917　38730972　38730976　38730977
传真：(020)38730925
网址：www.zlgmcu.com

广州专卖店

地址：广州市天河区新赛格电子城 203-204 室
电话：(020)87578634　87569917
传真：(020)87578842

南京周立功

地址：南京市珠江路 280 号珠江大厦 2006 室
电话：(025)83613221　83613271　83603500
传真：(025)83613271

北京周立功

地址：北京市海淀区知春路 113 号银网中心 A 座 1207-1208 室（中发电子市场斜对面）
电话：(010)62536178　62536179　82628073
传真：(010)82614433

重庆周立功

地址：重庆市石桥铺科园一路二号大西洋国际大厦（赛格电子市场）1611 室
电话：(023)68796438　68796439
传真：(023)68796439

杭州周立功

地址：杭州市天目山路 217 号江南电子大厦 502 室
电话：(0571)28139611　28139612　28139613　28139615　28139616　28139618
传真：(0571)28139621

成都周立功

地址：成都市一环路南二段 1 号数码同人港 401 室（磨子桥立交西北角）
电话：(028)85439836　85437446
传真：(028)85437896

深圳周立功

地址：深圳市深南中路 2070 号电子科技大厦 C 座 4 楼 D 室
电话：(0755)83781788（5 线）
传真：(0755)83793285

武汉周立功

地址：武汉市洪山区广埠屯珞瑜路 158 号 12128 室（华中电脑数码市场）
电话：(027)87168497　87168297　87168397
传真：(027)87163755

上海周立功

地址：上海市北京东路 668 号科技京城东座 7E 室
电话：(021)53083452　53083453　53083496
传真：(021)53083491

西安办事处

地址：西安市长安北路 54 号太平洋大厦 1201 室
电话：(029)87881296　83063000　87881295
传真：(029)87880865

销售与服务网络（二）

广州致远电子有限公司



地址：广州市天河区车陂路黄洲工业区 3 栋 2 楼
邮编：510660
传真：(020)38601859
网址：www.embedtools.com （嵌入式系统事业部）
www.embedcontrol.com （工控网络事业部）
www.ecardsys.com （楼宇自动化事业部）

技术支持：

CAN-bus：

电话：(020)22644381 22644382 22644253
邮箱：can.support@embedcontrol.com

iCAN 及数据采集：

电话：(020)28872344 22644373
邮箱：ican@embedcontrol.com

MiniARM：

电话：(020)28872684 28267813
邮箱：miniarm.support@embedtools.com

以太网：

电话：(020)22644380 22644385
邮箱：ethernet.support@embedcontrol.com

无线通讯：

电话：(020) 22644386
邮箱：wireless@embedcontrol.com

串行通讯：

电话：(020)28267800 22644385
邮箱：serial@embedcontrol.com

编程器：

电话：(020)22644371
邮箱：programmer@embedtools.com

分析仪器：

电话：(020)22644375 28872624 28872345
邮箱：tools@embedtools.com

ARM 嵌入式系统：

电话：(020)28872347 28872377 22644383 22644384
邮箱：arm.support@zlgmcu.com

楼宇自动化：

电话：(020)22644376 22644389 28267806
邮箱：mjs.support@ecardsys.com
mifare.support@zlgmcu.com

销售：

电话：(020)22644249 22644399 22644372 22644261 28872524
28872342 28872349 28872569 28872573 38601786

维修：

电话：(020)22644245

目　录

1. 功能简介

MUR-100 卡片读写器是广州致远电子自主开发的支持 IS014443-4 协议的新型卡片读写器，与动态库搭配可以完成多 NXP 新推出的 Mifare PLUS 卡片的读写操作。

1.1 产品功能特征

- 采用最新 PHILIPS 高集成 IS014443A 读卡芯片—MF RC500
- 采用最新 PHILIPS 高集成 PDIUSBD12，符合 USB1.1 协议
- USB 总线供电，整机电流小于 120mA
- 操作距离可达 9cm
- 未与 PC 机连接时，可指示感应区内是否有卡
- 提供丰富的 PC 机接口函数和演示程序
- 有蜂鸣器及发光二极管进行报警

1.2 产品外观



实际尺寸：124mm（长）×98mm（宽）×31mm（高）

2. MUR-100 工作状态说明

将 USB 插头插入 USB—B 型插座，读卡器即可上电，此时可通过观察读卡器上 LED 显示，判断读卡器所处状态。正常情况下，上电时红灯和绿灯同时点亮，约半秒钟后，绿灯灭，红灯继续亮，此时读卡器进入工作状态，可启动 PC 机程序对其进行操作；读卡器上电稳定后，红、绿灯的任何其它可能的状态组合均为故障。

一、红灯的作用

红灯是用来指示 Config()函数的执行状态的，若该函数执行成功，则红灯亮，否则红灯不亮。若读卡器执行了 Close() 函数，则红灯灭。

二、绿灯的作用

有四种情况可能改变绿灯的状态：

1、执行 Config()函数和 Close()函数

若 Config()函数执行成功，则绿灯灭，否则绿灯亮。若读卡器执行了 Close() 函数，则绿灯灭。

读卡器上电时，将自动执行 Config()函数，因此读卡器稳定后，若红灯亮、绿灯灭，则工作正常；若红灯灭、绿灯亮，则 RC500 初始化失败，将不能对卡进行操作，必须送修。

2、当读卡器上电进入正常工作状态后（此时红灯亮、绿灯灭），在 PC 机未发出任何命令之前，若读卡器感应区内有卡，绿灯将亮，否则绿灯灭。在 PC 机已发出任何一条命令后，此功能消失。因此在未启动 PC 机程序之前，本读卡器可以指示感应区内是否有卡。

3、执行 Set_Control_Bit()函数将熄灭绿灯，执行 Clr_Control_Bit()函数将点亮绿灯。

4、执行 Alarm()函数将可使绿灯闪烁。

三、蜂鸣器

执行 Alarm()函数可使直流蜂鸣器发声，可控制蜂鸣器发声的持续时间、间歇时间及重复次数。

四、USB 指示

读卡器一上电，当 USB 通信正常时，透过读卡器左边的百叶孔，可以看见内部有一绿色发光管点亮，当 USB 正在通信时该绿色发光管会闪烁。若该发光管不亮，则说明通信出错或 USB 驱动程序未安装。驱动安装方法请参考光盘附带的《USB 驱动安装说明》。

3. MUR—100 读卡器数据传输协议

3.1 数据块格式

电脑 USB 接口与 MUR-100 之间的通信协议说明。

3.1.1 主机→USB 读卡器（命令模式）

SeqNr	*Command*	*Len*	*Data[0…N]*	*BCC*

INFO[0] . . . . . . INFO[n]

SeqNr： 1 Byte 数据交换包的序号

Command： 1 Byte 命令字符

Len： 1 Byte 数据的长度

Data[…]： *Len* Byte 数据字节

BCC： 1Byte 的 BCC 校验

3.1.2 USB 读卡器→主机（响应模式）

SeqNr	*status*	*Len*	*Data[0…N]*	*BCC*

INFO[0] . . . . . . INFO[n]

SeqNr： 1 Byte 数据交换包的序号

status： 1 Byte 状态字符

Len： 1 Byte 数据的长度

Data[…]： *Len* Byte 数据字节

BCC： 1Byte 的 BCC 校验

3.1.3 数据块格式描述

- 数据交换包的序号由主机发送数据块时产生，取值范围为 0-255。在经过一次正确的数据交换后，主机在发送下一个命令时，将数据包的序号加 1。读卡器返回最近接收的包序号。通常主机应用程序最好检查命令/响应包交换时的数据包的序号。
- 不管在执行命令时出现了任何错误，响应包中的数据长度为 0（*Len* = 0）。
- BCC 校验码计算数据块中所有的 INFO 字节。然后将结果传送到数据块的最后一个字节，如下式所示：

  INFO[n] = BCC = ~（INFO[0] ⊕ INFO[1] ⊕ ... ⊕ INFO[n-1]） (⊕ ... XOR、~…NOT)

3.2 主机命令与参数

3.2.1 底层函数和高级函数

命令		参数		补充说明
名称	数值	发送	接收	
Request	0x41	*_Mode*	*_TagType*	发出询问命令，检查在有效范围内是否有卡存在
Anticoll	0x42	*_Bcnt*	*_SNR*	开始防冲突操作，返回卡的序号
Anticoll2	0x71	*_Encoll,_Bcnt*	*_SNR*	可禁止或允许多张卡进入
Select	0x43	*_SNR*	*_Size*	选择卡，返回卡的存贮容量
Authentication	0x44	*_Mode,_SecNr*	--	用 EEPROM 中的密码验证
Authentication2	0x72	*_Mode,_SecNr,_KeyNr*	--	选择密匙区 E2 中的密码区验证
AuthKey	0x73	*_Mode,_SecNr,_Key(6)*	--	直接密码验证
Halt	0x45	--	--	将卡置于挂起模式
Read	0x46	*_Adr*	*_Data*	从卡中相应地址中读出一个 16 字节的块
Write	0x47	*_Adr,_Data*	--	向卡中相应地址写入一 16 字节的数据块
Value	0x70	*_Mode,_Adr,_Value,_Trans_Adr*	--	包含加、减、恢复函数，并带自动传送
LoadKey	0x4C	*_Mode,_SecNr,,_Nkey*	--	改变存贮在 EEPROM 中的密钥
Reset	0x4E	*_Msec*	--	关闭天线输出数 ms，使卡复位
Get Info	0x4F	--	*_Info*	读取固件信息 RC500 序列号
Set Control Bit	0x50	--	--	将控制位置为高电平
Clr Control Bit	0x51	--	--	将控制位置为低电平
Config	0x52	--	--	复位且配置 RC500
Close	0x3F	--	--	关闭 RC500
Check Write	0x53	*_SNR,_Authmode,_Adr,_Data*	--	将所传送的数据和上一次所写的数据进行比较
Alarm	0x60	*_Control,_Opentm,* *_Closetm,_Repcnt*	--	输出控制信号，能控制动作时间、间隙时间和重复次数
Read E2	0x61	*_Adr,_Length*	*_Data*	读 RC500 内 EEPROM 的内容
Write E2	0x62	*_Adr,_Length,_Data*	--	写数据到 RC500 内 EEPROM

华虹 SHC1102 卡操作函数				
SHC1102_Auth	0x80	*_KeyBlock, *_Key*		4 字节密码验证
SHC1102_Read	0x81	*_Block*	**_Data*	读 4 字节数据
SHC1102_Write	0x82	*_Block, *_Data*		写 4 字节数据
ISO14443-4 协议支持函数				
PiccExChangBlock	0x90	*ucCRC，ucFWI，*pSendBfr, usSendBit,*	**pRcvBfr, *pRcvBit, *pTime*	*PCD 与 PICC 交换数据块
PiccRATS	0x91	*ucCID*	**pATS*	*选择应答，获取卡片信息
PiccPPS	0x92	*ucDSI， ucDRI*	--	*协议和参数选择请求
PiccDeSelect	0x93	--	--	*解除激活
PiccTPCL	0x94	**pSBfr, usSLen*	**pRBfr, *pRLen*	*T=CL 协议，直接实现读卡器到卡片的数据分组透传

以上补充说明栏中，打“*”号的是此版本的新增函数。

3.2.2 版本升级说明

- 2003 年 9 月 30 日，在 Get_Info()的返回数据中增加了一字节的固件版本号。函数 LoadKey、Authentication 和 Authentication2 增加了对密钥区号必须小于 16 的限制。对 Mifare1 S70 卡的支持更加完善。
- 2003 年 10 月 16 日，增加了对华虹 SHC1102 卡的操作支持。
- 2009 年 08 月 06 日，增加了对 ISO14443-4 协议的支持。通过上位机动态库内的函数可以实现对 PLUS CPU 卡的操作。

3.2.3 状态值列表

名称	值	描述
MI_OK, COMM_OK	0	函数调用成功
MI_NOTAGERR	1	在有效区域内没有卡
MI_CRCERR	2	从卡中接收到了错误的 CRC 校验和
MI_EMPTY	3	值溢出
MI_AUTHERR	4	不能验证
MI_PARITYERR	5	从卡中接收到了错误的校验位
MI_CODEERR	6	编码错误
MI_SENDRERR	8	在防冲突时读到了错误的串行码
MI_KEYERR	9	证实密码错
MI_NOTAUTHERR	10	卡没有验证
MI_BITCOUNTERR	11	从卡中接收到了错误数量的位
MI_BYTECOUNTERR	12	从卡中接收了错误数量的字节
MI_TRANSERR	14	调用 Transfer 函数出错
MI_WRITEERR	15	调用 Write 函数出错
MI_INCRERR	16	调用 Increment 函数出错
MI_DECRERR	17	调用 Decrment 函数出错
MI_READERR	18	调用 Read 函数出错
MI_COLLERR	24	冲突错
MI_ACCESSTIMEOUT	27	访问超时
MI_QUIT	30	上一次了送命令时被打断
MIS_CHK_OK	0	Check Write 正确
MIS_CHK_FAILED	1	Check Write 出错
MIS_CHK_COMPERR	2	Check Write:写出错（比较出错）
COMM_ERR	255	串行通信错误

3.3 函数描述

下面的描述可以作为函数声明。

3.3.1 请求—Request

声明：

uchar mifs_request(uchar _Mode,uchar idata *_TagType);

主机 ⇒ 读卡器：

命令符： 0x41

长度： 1

Data[0]: _Mode

读卡器 ⇒ 主机 ：

状态值： MI_OK ， MI_QUIT ， MI_NOTAGERR ， MI_BITCOUNTERR ，COMM_ERR

长度： 2

Data[0]: tagtype (低字节)

Data[1]: tagtype (高字节)

参数：

_Mode：

							ALL

ALL=0：请求天线范围内 IDLE 状态的卡（HALT 状态的除外）

ALL=1：请求天线范围内的所有卡

_tagtype： 当发生错误时，不返回任何内容（Len=0）

描述：

此函数发送 Request 命令，检查在有效范围内是否有卡存在。这个函数在选择一个新的卡，是必须调用的。

3.3.2 防碰撞—Anticoll

声明：

uchar mifs_anticoll(uchar _Bcnt,uchar idata *_SNR);

主机 ⇒ 读卡器：

命令符： 0x42

长度： 1

Data[0]： _Bcnt

读卡器 ⇒ 主机：

状态值： MI_OK ， MI_QUIT ， MI_NOTAGERR ， MI_BITCOUNTERR ， COMM_ERR

长度： 4

Data[0]： snr(LL)

Data[1]： snr(LH)

Data[2]： snr(HL)

Data[3]： snr(HH)

参数：

_Bcnt： 为预选卡所分配的位的个数，通常Bcnt=0

_SNR： 卡的序列号。存贮在一个无符号的四字节数组中，低字节放在地址处。

描述：

此函数开始防冲突操作。必须在调用了Request命令后立即调用。当知道了所要选择卡的序列号后，就没有必要调用AntiColl。此时，调用了Request后，直接调用Select函数即可。

3.3.3 选择—Select

声明：

```
uchar mifs_select(uchar idata *_SNR,uchar idata *_Size);
```

主机 ⇒ 读卡器：

命令符： 0x43
长度： 4
Data[0]： snr(LL)
Data[1]： snr(LH)
Data[2]： snr(HL)
Data[3]： snr(HH)

读卡器 ⇒ 主机：

状态值： MI_OK, MI_QUIT, MI_NOTAGERR, MI_CRCERR, MI_PARITYERR, MI_BITCOUNTERR,COMM_ERR
长度： 1
Data[0]： _Size

参数：

_SNR： 卡的序号。存贮在一个无符号4字节字符数组中，低字节放在代地址处。
_Size： 当Select命令返回值为MI_OK时，ATS (answer to select)将返回主机。

描述：

这个函数选择某一个序列号的卡，返回ATS字节给主机。

3.3.4 证实—Authentication

声明：

uchar mifs_authentication(uchar _Mode,uchar _SecNr);

主机 ⇒ 读卡器：

命令符：0x44

长度：2

Data[0]：_Mode

Data[1]：_SecNr

读卡器 ⇒ 主机：

状态值：MI_OK, MI_QUIT, MI_NOTAGERR, MI_AUTHERR,MI_BITCOUNTERR, MI_PARITYERR, COMM_ERR

长度：0

参数：

_Mode：

					AB		

AB = 0：利用密钥A进行验证

AB = 1：利用密钥B进行验证

_SecNr：所访问卡的扇区号，必须小于16。

描述：

在对卡进行读、写、加、减等操作前，必须对卡进行验证。若卡中一扇区的密钥与RC500中相应密码区存储的密码相匹配。则证实成功，函数将返回MI_OK。

3.3.5 暂停—Halt

声明：

uchar mifs_halt(void);

主机 ⇒ 读卡器：

命令符： 0x45

长度： 0

读卡器 ⇒ 主机：

状态值： MI_OK，MI_QUIT，MI_CODE， COMM_ERR

长度： 0

参数：

无

描述：

此函数将所选择卡置为挂起状态。如果要进行重新选择，则应用ALL模式调用Request命令。或将卡复位，如将卡离开天线操作区再进入，或执行复位函数mifs_reset();

3.3.6 读—Read

描述：

uchar mifs_read(uchar _Adr,uchar idata *_Data);

主机 ⇒ 读卡器：

命令符：0x46

长度：1

Data[0]：_Adr

读卡器 ⇒ 主机：

状态值：MI_OK，MI_QUIT，MI_NOTAGERR，MI_CRCERR，MI_NOTAUTHERR，MI_PAROTUERR，MI_BITCOUNTERR，COMM_ERR

长度：16

Data[0]：所访问块的第一个字节

：

Data[15]：所访问块的最后一个字节

参数：

_Adr：所读数据地址

描述：

此函数在所选的卡通过验证后，读取一个16字节的块。

3.3.7 写—Write

描述：

uchar mifs_write(uchar _Adr, uchar idata *_Data);

主机 ⇒ 读卡器：

命令符：　　0x47

长度：　　17

Data[0]：　　address

Data[1]：　　所访问块的第一个字节

：

Data[16]：　　所访问块的最后一个字节

读卡器 ⇒ 主机：

状态值：　　MI_OK，MI_QUIT，MI_NOTAGERR，MI_NOTAUTHERR，MI_WRITEERR，MI_BITCOUNTERR，COMM_ERR

长度：　　0

参数：

_Adr：　　所写数据块地址（0—63）

_Data：　　16 字节数据指针

描述：

此函数在所选的卡通过验证后，写入一个16字节的块。

3.3.8 装载密钥—Load Key

声明：

uchar mifs_load_key(uchar _Mode,uchar _SecNr,uchar *_Nkey);

主机 ⇒读卡器：

命令符： 0x4C

长度： 8

Data[0]： _Mode

Data[1]： _SecNr

Data[2]： _Nkey[0]

: :

Data[7]： _Nkey[5]

读卡器⇒主机：

状态值： MI_OK，MI_QUIT，MI_AUTHERR，COMM_ERR

长度： 0

参数：

_Mode：

					AB		

AB = 0：利用密钥A进行验证

AB = 1：利用密钥B进行验证

_SecNr: 密钥扇区号，必须要小于 16。

_Nkey: 6 字节密钥首址

描述：

这个函数将一个新的密钥写入到 RC500 的只写 EEPROM 存贮器的相应区中。

3.3.9 复位—Reset

声明：

uchar mifs_reset(uchar _Msec);

主机 ⇒读卡器：

命令符： 0x4E

长度： 1

Data[0]： _Msec

读卡器⇒主机：

状态值： MI_OK，MI_QUIT，COMM_ERR

长度： 0

参数：

_Msec： 射频电路关闭时间（以毫秒为单位）

描述：

该函数使射频电路关闭所规定的时间，若_Msec=0，射频电路部分将一直处于关闭状态，一直到下一个 Request 命令到来。关闭射频能使天线内的所有卡复位。

举例：

_Msec = 0	⇒	∞	射频电路关闭
_Msec = 0x01	⇒	1 ms	射频电路关闭 1ms
_Msec= 0xFF	⇒	255 ms	射频电路关闭 255ms

3.3.10 获取信息—Get Info

声明：

uchar mifs_get_info(uchar idata *_Info);

主机 ⇒读卡器：

命令符： 0x4F

长度： 0

读卡器⇒主机：

状态值： MI_OK，MI_QUIT，COMM_ERR

长度： 9（ver1.0）,10（ver1.1 以上）

Data[0]： 产品类型标识 0

: :

Data[4]： 产品类型标识 4

Data[5]： RC500 序列号 0

: :

Data[8]： RC500 序列号 3

Data[9]： 固件版本号（ver1.1 以上）

参数：

_Info： _Info[0]—_Info[4]为 RC500 的产品类型标识，依次为 0x30,0x88,0xf8,0x00,0xXX

_Info[5]—_Info[8]为 RC500 的序列号

_Info[9]为固件版本号，Ver1.1 以上的版本才有，高四位为版本号的整数，取值从 1 到 15，低四位为版本号的小数，取值从 0 到 9。

描述：

此函数返回一个包含有 RC500 的产品类型标识、序列号和固件版本号的数组。

3.3.11 置位控制位—Set Control Bit

声明：

uchar mifs_set_control_bit();

主机 ⇒读卡器：

命令符： 0x50

长度： 0

读卡器⇒主机：

状态值： MI_OK，MI_QUIT，COMM_ERR

长度： 0

描述：

此函数设置 MIFARE 读卡器中的控制位为高电平。

3.3.12 清除控制位—Clr Control Bit

声明：

```
uchar mifs_clr_control_bit();
```

主机 ⇒读卡器：

命令符： 0x51

长度： 0

读卡器⇒主机：

状态值： MI_OK，MI_QUIT，COMM_ERR

长度： 0

描述：

此函数清除 MIFARE 读卡器中的控制位

3.3.13 配置—Config

声明：

uchar mifs_config(void);

主机 ⇒读卡器：

命令符： 0x52

长度： 0

读卡器⇒主机：

状态值： MI_OK，MI_QUIT，COMM_ERR

长度： 0

参数：

说明：

RC500 每次上电复位之后，都必须首先调用此函数对模块进行初始化，才能进行进一步的操作。

3.3.14 检查写—Check Write

声明：

uchar mifs_check_write(uchar idata *_SNR, uchar _Authmode, uchar _Adr, uchar idata *_Data);

主机 ⇒读卡器：

命令符： 0x53

长度： 22

Data[0]： _SNR(LL)

Data[1]： _SNR(LH)

Data[2]： _SNR(HL)

Data[3]： _SNR(HH)

Data[4]： _Authmode

Data[5]： _Adr

Data[6]： 块的第一个字节

::

Data[21]： 块的最后一个字节

读卡器⇒主机：

状态值： MI_QUIT，MIS_CHK_OK，MIS_CHK_FAILED，MIS_CHK_COMPERR，COMM_ERR

长度： 0

参数：

_SNR： 所要检查的卡的序号

_Authmode： 上一次写命令时的验证模式

_Adr： 所要检查的数据块的地址

_Data： 所检查的数据

描述：

此函数在数据写入卡的数据进行检查。将重新进行 Request/Select/Authenticated 操作。此函数进行将所给出的数据与相应地址的数据进行比较。如果正确，则返回 MIS_CHK_OK 信息。如果两者间数据不相符，则返回 MIS_CHK_COMPERR 信息。发生其它任何错误时，返回 MIS_CHK_FAILED 信息。

注：验证密匙时所用的密匙区与块_Adr 所在的扇区号相同。

3.3.15 输出报警信号—Alarm

声明：

mifs_Alarm(uchar _Contrl, uchar _Opentm, uchar _Closetm, uchar _Repcnt);

主机 ⇒读卡器：

命令符： 0x60

长度： 2

Data[0]： _Control

Data[1]: _Opentm

Data[2]: _Closetm

Data[3]: _Repcnt

读卡器⇒主机：

状态值： MI_OK， COMM_ERR

长度： 0

参数：

_Control: 控制字，如下表相应位为 1 时该器件动作。

						绿灯	蜂鸣器

_Opentm: 低电平持续时间，取值（0—255），10ms 的分辨率

_Closetm: 高电平间隙时间，取值（0—255），10ms 的分辨率

_Repcnt: 重复次数

描述：

此函数输出一驱动信号可驱动蜂鸣器和绿色发光管，持续时间、间隙时间和重复次数可调。命令发出后主机可立即收到响应，但由于读卡器正处于报警状态，因此要等到报警完毕后才可发送下一条命令。

3.3.16 读 EEPROM

声明：

uchar mifs_read_E2(uchar _Adr,uchar _Length,uchar idata *_Data);

主机 ⇒读卡器：

命令符： 0x61

长度： 2

Data[0]： _Adr

Data[1]： _Length

读卡器⇒主机：

状态值： MI_OK，MI_QUIT，MI_CRCERR，MI_BITCOUNTERR，COMM_ERR

长度： _Length

Data[0]： byte

…

Data[_Length]：byte

参数：

_Adr： 被读 RC500 内 EEPROM 首址，必须小于 0x80

_Length： 被读数据长度

_Data： 读出数据缓冲区首址

描述：

此函数将 RC500 内 EEPROM 的数据读出。

3.3.17 写 EEPROM

声明：

```
uchar mifs_write_E2(uchar _Adr,uchar _Length,uchar idata *_Data);
```

主机 ⇒读卡器：

命令符： 0x62

长度： _Length+2

Data[0]： _Adr

Data[1]： _Length

Data[2]： _Data[0]

…

Data[_Length+1]： _Data[_Length-1]

读卡器⇒主机：

状态值： MI_OK，MI_QUIT，MI_CRCERR，MI_BITCOUNTERR，COMM_ERR

长度： 0

参数：

_Adr： RC500 内 EEPROM 的写入首址，取值范围（0x30—0x7E）

_Length： 被写数据长度

_Data： 写入数据缓冲区首址

描述：

此函数将数据写入 RC500 内 EEPROM 中，地址范围为 0x30—0x7F。RC500 内 EEPROM 的 0x00—0x0F 为只读产品信息区，0x10—0x2F 为启动寄存器初始化文件区，最好不要改写，0x80—0x1FF 为只读密钥区，可用 LoadKey 写入。

3.3.18 关闭 RC500—Close

声明：

uchar mifs_close(void);

主机 ⇒读卡器：

命令符： 0x3F

长度： 0

读卡器⇒主机：

状态值： MI_OK， COMM_ERR

长度： 0

参数：

描述：

此函数将 RC500 的复位管脚置为高电平，关闭 RC500，使之电流最小。若要重新启动则需调用 Config()。

3.3.19 值操作

声明：

uchar mifs_value(uchar _*Mode, uchar _Adr, ulong idata *_Value, uchar _Trans*_Adr);

主机 ⇒读卡器：

命令符： 0x70

长度： 7

Data[0]： _Mode

Data[1]： _Adr

Data[2]： _Value(LL)

Data[3]： _Value(LH)

Data[4]： _Value(HL)

Data[5]： _Value(HH)

Data[6]： _Trans_Adr

读卡器⇒主机：

状态值： MI_OK, MI_QUIT, MI_NOTAGERR, MI_CODE, MI_BITCOUNTERR, MI_TRANSERR, MI_CODEERR, COMM_RERR

长度： 0

参数：

_Mode: 0xC0—减
0xC1—加
0xC2—恢复

_Adr： 卡内块地址，对该块进行值操作，取值范围 0—63。

_Value： 当进行加或减操作时，为加数或减数；当进行恢复操作时，该值为空值。

_Trans_Adr： 传输块地址，取值范围 0—63。

描述：

此函数对卡内的某一块进行加、减或数据备份，该块必须为值块格式，并支持自动传送。

3.3.20 防碰撞 2—Anticoll2

声明：

uchar mifs_anticoll2(uchar _Encoll, uchar _Bcnt,uchar idata *_SNR);

主机 ⇒ 读卡器：

命令符： 0x71

长度： 2

Data[0]: _Encoll

Data[1]: _Bcnt

读卡器 ⇒ 主机：

状态值： MI_OK，MI_NOTAGERR，MI_BITCOUNTERR，MI_COLLERR，COMM_ERR

长度： 4

Data[0]: snr(LL)

Data[1]: snr(LH)

Data[2]: snr(HL)

Data[3]: snr(HH)

参数：

_Encoll: 若为1，则使能多张卡进入天线区；若为0，则不多张卡进入，此时返回错误MI_COLLERR.

_Bcnt： 为预选卡所分配的位的个数，通常Bcnt=0

_SNR： 卡的序列号。存贮在一个无符号的四字节数组中，低字节放在地址处。

描述：

此函数开始防冲突操作。必须在调用了Request命令后立即调用。当知道了所要选择卡的序列号后，就没有必要调用AntiColl。此时，调用了Request后，直接调用Select函数即可。

3.3.21 证实 2—Authentication2

声明：

uchar mifs_authentication2(uchar _Mode, uchar _SecNr, uchar _KeyNr);

主机 ⇒ 读卡器：

命令符： 0x72

长度： 3

Data[0]： _Mode

Data[1]： _SecNr

Data[2]： _KeyNr

读卡器 ⇒ 主机：

状态值： MI_OK ， MI_QUIT ， MI_NOTAGERR ， MI_PAROTUERR ， MI_BITCOUNTERR，COMM_ERR

长度： 0

参数：

_Mode：

					AB		

AB = 0：利用密钥A进行验证

AB = 1：利用密钥B进行验证

_SecNr： 所访问卡的扇区号

_KeyNr： 用于证实的密匙区号，必须小于16。

描述：

在对卡进行读、写、加、减等操作前，必须对卡进行验证。若卡中的密钥与RC500中所选择的密码相匹配。则证实成功，函数将返回MI_OK。

3.3.22 直接密码证实—AuthKey

声明：

uchar mifs_authentication2(uchar _Mode, uchar *_Key, uchar _SecNr);

主机 ⇒ 读卡器：

命令符： 0x73

长度： 8

Data[0]： _Mode

Data[1]： _SecNr

Data[2]： _Key[0]

…

Data[7]： _Key[5]

读卡器 ⇒ 主机：

状态值： MI_OK，MI_QUIT，MI_NOTAGERR，MI_PAROTUERR，MI_BITCOUNTERR，COMM_ERR

长度： 0

参数：

_Mode：

					AB		

AB = 0：利用密钥A进行验证

AB = 1：利用密钥B进行验证

_SecNr： 所访问卡的扇区号

_Key： 用于证实的密码首址

描述：

在对卡进行读、写、加、减等操作前，必须对卡进行验证。若卡中的密钥与所传输的密码相匹配。则证实成功，函数将返回MI_OK。

3.3.23 华虹 1102 验证

声明：

uchar SHC1102_Auth(uchar *_KeyBlock, uchar* *_Key);

主机 ⇒读卡器：

命令符：　　0x80

长度：　　5

Data[0]：　　_KeyBlock

Data[1]：　　_Key[0]

Data[2]：　　_Key[1]

Data[3]：　　_Key[2]

Data[4]：　　_Key[3]

读卡器⇒主机：

状态值：　　MI_OK, MI_QUIT, MI_NOTAGERR, MI_CODE, MI_BITCOUNTERR, MI_TRANSERR, MI_CODEERR, COMM_RERR

长度：　　0

参数：

_KeyBlock：　　密码块地址，对于 SHC1102 为 8

_Key：　　4 字节密码首址

描述：

此函数将参数中的密码与卡中密码块的密码进行比较，若密码匹配则函数返回 MI_OK。

3.3.24 华虹 1102 读

声明：

uchar SHC1102_Read(uchar *_Block, uchar *_Data*);

主机 ⇒读卡器：

命令符： 0x81

长度： 1

Data[0]： _Block

读卡器⇒主机：

状态值： MI_OK, MI_QUIT, MI_NOTAGERR, MI_CODE, MI_BITCOUNTERR, MI_TRANSERR, MI_CODEERR, COMM_RERR

长度： 4

Data[0]： _Data[0]

Data[1]： _Data[1]

Data[2]： _Data[2]

Data[3]： _Data[3]

参数：

_Block： 所读块地址

_Data： 4 字节数据首址

描述：

此函数将卡中的数据块读出。

3.3.25 华虹 1102 写

声明：

uchar SHC1102_Write(uchar *_Block, uchar *_Data*);

主机 ⇒读卡器：

命令符： 0x82

长度： 5

Data[0]： _Block

Data[0]： _Data[0]

Data[1]： _Data[1]

Data[2]： _Data[2]

Data[3]： _Data[3]

读卡器⇒主机：

状态值： MI_OK, MI_QUIT, MI_NOTAGERR, MI_CODE, MI_BITCOUNTERR, MI_TRANSERR, MI_CODEERR, COMM_RERR

长度： 0

参数：

_Block： 所写块地址

_Data： 4 字节数据首址

描述：

此函数将数据写入卡中。

3.3.26 ISO14443-4 数据块交换

声明：

unsigned char __stdcall PiccExChangBlock(unsigned char ucCRC, unsigned char ucFWI,unsigned char *pSendBfr,unsigned short usSendBit,unsigned char *pRcvBfr, unsigned short *pRcvBit,int *pTime)

主机 ⇒读卡器：

命令符： 0x90

长度： usSendBit/8 + 2

Data[0]： ucCRC

Data[1]： ucFWI

Data[2]： pSendBfr[0]

……

Data[usSendBit/8+1]： pSendBfr[usSendBit/8-1]

读卡器⇒主机：

状态值： MI_OK, MI_QUIT, MI_NOTAGERR, MI_CODE, MI_BITCOUNTERR, MI_TRANSERR, MI_CODEERR, COMM_RERR

长度： pRcvBit /8+2

Data[0]： _ ucCRC

Data[1]： _ pTime 计算因子

Data[2]： pRcvBfr [0]

Data[3]： pRcvBfr [1]

……

Data[pRcvBit /8 + 1]： pRcvBfr [pRcvBit /8+2 – 1]

参数：

```
unsigned char ucCRC;          b1 = 1,发送使能CRC;b1 = 0,发送禁止CRC;
                              b0 = 1,接收使能CRC;b0 = 0,接收禁止CRC;
unsigned char ucFWI          ; 超时等待时间编码，(0～17)
unsigned char *pSendBfr      ;发送数据缓冲区首址
unsigned short usSendBit     ; 发送数据的位数(小于等于56字节)
unsigned char *pRcvBfr       ;接收数据缓冲区首址
unsigned short *pRcvBit      ; 接收数据的位数
int *pTime                   ; PCD发送数据到PICC回应的时间(us)
```

描述：

pTime = (pTime 计算因子 (0x01 << ucFWI))* 10000/8475; // 单位 us

3.3.27 ISO14443-4 选择应答

声明：

```
unsigned char __stdcall PiccRATS(unsigned char ucCID, unsigned char *pATS)
```

主机 ⇒读卡器：

命令符： 0x91

长度： 1

Data[0]： ucCID

读卡器⇒主机：

状态值： MI_OK, MI_QUIT, MI_NOTAGERR, MI_CODE, MI_BITCOUNTERR, MI_TRANSERR, MI_CODEERR, COMM_RERR

长度： Data[0]

Data[0]： pATS [0]

Data[1]： pATS [1]

Data[2]： pATS [2]

Data[3]： pATS [3]

……

参数：

unsigned char ucCID ； 卡标识符(0 – 0x0E)

unsigned char *pATS ； 返回的ATS信息

描述：

返回的数据长度为pATS[0]

3.3.28 ISO14443-4 协议和参数选择请求

声明：

```
unsigned char __stdcall PiccPPS(unsigned char ucDSI, unsigned char ucDRI)
```

主机 ⇒读卡器：

命令符： 0x92

长度： 0

读卡器⇒主机：

状态值： MI_OK, MI_QUIT, MI_NOTAGERR, MI_CODE, MI_BITCOUNTERR, MI_TRANSERR, MI_CODEERR, COMM_RERR

长度： 0

参数：

unsigned char ucDSI	PICC 到 PCD的波特率(0-3)
unsigned char ucDRI	PCD 到 PICC的波特率(0-3)

描述：

MUR-100不支持动态更改通信波特率，固定为106Kbit/s

3.3.29 ISO14443-4 解除激活

声明:

```
unsigned char __stdcall PiccDeSelect(void)
```

主机 ⇒读卡器：

命令符： 0x93

长度： 0

读卡器⇒主机：

状态值： MI_OK, MI_QUIT, MI_NOTAGERR, MI_CODE, MI_BITCOUNTERR, MI_TRANSERR, MI_CODEERR, COMM_RERR

长度： 0

3.3.30 ISO14443-4 T=CL 数据透传分组传输指令

声明：

```
unsigned char __stdcall PiccTPCL(unsigned char *pSBfr, unsigned short usSLen,
                                  unsigned char *pRBfr, unsigned short *pRLen)
```

主机 ⇒读卡器：

命令符： 0x94

长度： usSLen 小于 CPU 卡支持的单次 USB 传输最大字节数时长度为 usSLen

usSLen 大于 CPU 卡支持的单次 USB 传输最大字节数时长度为**单次传输最大字节数**

Data[0]： pSBfr [0]

Data[1]： pSBfr [1]

Data[2]： pSBfr [2]

Data[3]： pSBfr [3]

……

读卡器⇒主机：

状态值： MI_OK, MI_QUIT, MI_NOTAGERR, MI_CODE, MI_BITCOUNTERR, MI_TRANSERR, MI_CODEERR, COMM_RERR

长度： pRLen 小于 CPU 卡支持的单次 USB 传输最大字节数时长度为 usSLen

pRLen 大于 CPU 卡支持的单次 USB 传输最大字节数时长度为**单次传输最大字节数**

Data[0]： pRBfr [0]

Data[1]： pRBfr [1]

Data[2]： pRBfr [2]

Data[3]： pRBfr [3]

……

参数：

```
unsigned char *pSBfr      ; 发送数据缓冲区
hort usSLen               ; 发送的字节数
unsigned char *pRBfr      ; 接收数据缓冲区
unsigned   short *pRLen   ; 接收的字节数
```